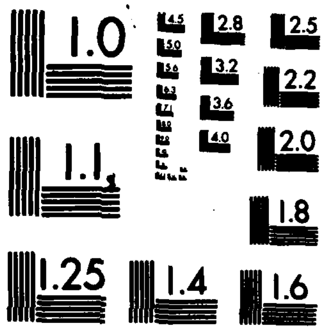
AD-A129 334 MULTI-ECHELON REPAIRABLE ITEM PROVISIONING IN A TIME-VARYING ENVIRONMENT..(U) GEORGE WASHINGTON UNIV WASHINGTON DC PROGRAM IN LOGISTICS D GROSS ET AL. 1/1

UNCLASSIFIED 22 OCT 82 SERIAL-T-468 N00014-75-C-0729 F/G 12/1 NL

END
DATE
FILMED
DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

AD A129334

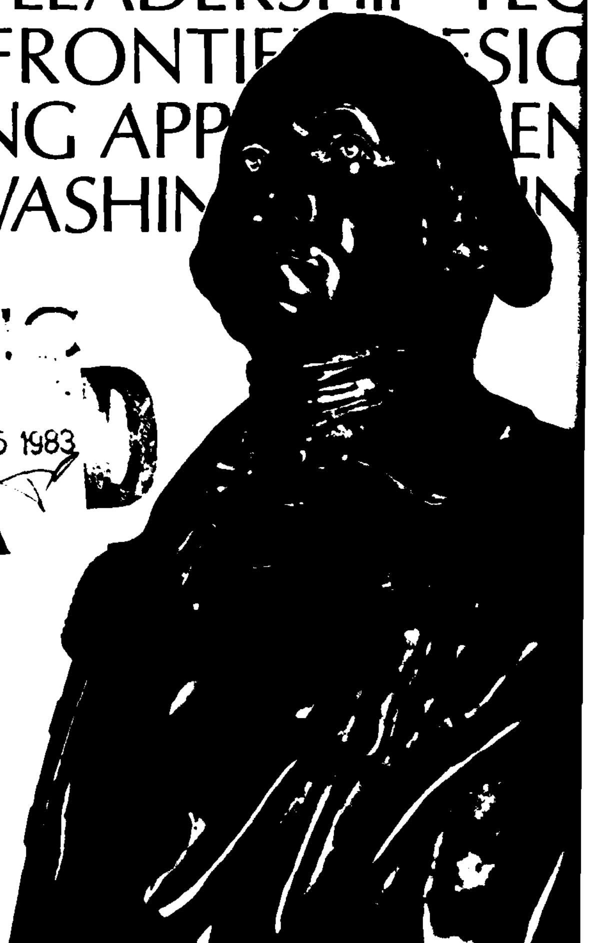
MULTI-ECHELON REPAIRABLE ITEM PROVISIONING
IN A TIME-VARYING ENVIRONMENT USING
THE RANDOMIZATION TECHNIQUE

by

Donald Gross*†
Douglas R. Miller*

THE
GEORGE
WASHINGTON
UNIVERSITY

STUDENTS FACULTY STUDY
ESEARCH DEVELOPMENT FU
URE CAREER CREATIVITY C
MMUNITY LEADERSHIP TEC
NOLOGY FRONTI ESIC
ENGINEERING APP EN
GEORGE WASHIN N



INSTITUTE FOR MANAGEMENT
SCIENCE AND ENGINEERING
SCHOOL OF ENGINEERING
AND APPLIED SCIENCE

MULTI-ECHELON REPAIRABLE ITEM PROVISIONING IN A TIME-VARYING ENVIRONMENT USING THE RANDOMIZATION TECHNIQUE

by

Donald Gross*†
Douglas R. Miller*

Serial T-468
22 October 1982

The George Washington University
School of Engineering and Applied Science
Institute for Management Science and Engineering

Research Sponsored by
*Air Force Logistics Management Center
Contract SCEEE/AFLMC/80-6
and
†Program in Logistics
Contract N00014-75-C-0729
Project NR 347 020
Office of Naval Research

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER T-468	2. GOVT ACCESSION NO. AD-A129334	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) MULTI-ECHELON REPAIRABLE ITEM PROVISIONING IN A TIME-VARYING ENVIRONMENT USING THE RANDOMIZATION TECHNIQUE		5. TYPE OF REPORT & PERIOD COVERED SCIENTIFIC
		6. PERFORMING ORG. REPORT NUMBER T-468
7. AUTHOR(s) DONALD GROSS, DOUGLAS R. MILLER		8. CONTRACT OR GRANT NUMBER(s) N00014-75-C-0729 SCEEE/AFLMC/80-6
9. PERFORMING ORGANIZATION NAME AND ADDRESS THE GEORGE WASHINGTON UNIVERSITY PROGRAM IN LOGISTICS WASHINGTON, DC 20052		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS OFFICE OF NAVAL RESEARCH CODE 411 ARLINGTON, VA 22217		12. REPORT DATE 22 OCTOBER 1982
		13. NUMBER OF PAGES 37
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

APPROVED FOR PUBLIC SALE AND RELEASE; DISTRIBUTION UNLIMITED.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

MULTI-ECHELON INVENTORY
TRANSIENT SOLUTIONS TO MARKOV PROCESSES
CLOSED QUEUEING NETWORKS

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

Multi-echelon repairable item provisioning systems are considered under a time-varying environment. Such conditions could arise, for example, in a military context where a shift from peacetime operation to wartime operation takes place; or, in a civilian setting where a public transit system decides to increase its hours of operation or frequency of service.

Exact Markovian models, incorporating a finite population of repairable

(Cont'd)

20. Abstract (cont'd)

components and limited repair capacity (nonsample service), are treated, with transient solutions obtained using the randomization technique.

THE GEORGE WASHINGTON UNIVERSITY
School of Engineering and Applied Science
Institute for Management Science and Engineering

Program in Logistics

Abstract
of
Serial T-468
22 October 1982

MULTI-ECHELON REPAIRABLE ITEM PROVISIONING IN A TIME-VARYING ENVIRONMENT USING THE RANDOMIZATION TECHNIQUE

by

Donald Gross
Douglas R. Miller

Multi-echelon repairable item provisioning systems are considered under a time-varying environment. Such conditions could arise, for example, in a military context where a shift from peacetime operation to wartime operation takes place; or, in a civilian setting where a public transit system decides to increase its hours of operation or frequency of service.

Exact Markovian models, incorporating a *finite* population of repairable components and *limited* repair capacity (nonample service), are treated, with transient solutions obtained using the randomization technique.

Research Sponsored by
Air Force Logistics Management Center
and
Office of Naval Research

THE GEORGE WASHINGTON UNIVERSITY
School of Engineering and Applied Science
Institute for Management Science and Engineering
Program in Logistics

MULTI-ECHELON REPAIRABLE ITEM PROVISIONING IN A TIME-VARYING ENVIRONMENT USING THE RANDOMIZATION TECHNIQUE

by

Donald Gross
Douglas R. Miller

1. INTRODUCTION

This paper develops models of the transient behavior of Markovian repairable item provisioning systems. A multi-echelon structure of repair and resupply is of concern. It is desired to develop analytical solution techniques for *exact* models of systems with *finite* numbers of items and *finite* repair capacities (i.e., a finite number of repair channels at each repair facility). This is in contrast to Dyna-METRIC [see Hillestad (1981) and Hillestad and Carrillo (1980)], which is an approximate model assuming an *infinite* calling population of items and *ample* repair capacities.

Most multi-echelon repairable item work has concentrated on steady state solutions and revolves around the METRIC type of model, which assumes an infinite population of items which can fail and an infinite repair capacity, so that no queue ever forms at the repair

facility [see Sherbrooke (1968) and Muckstadt (1973), for example]. Recently, the METRIC type of model has been extended to provide transient solutions for a time varying environment (Hillestad and Hillestad and Carrillo, *op. cit.*), and is called Dyna-METRIC.

Exact models for finite item populations and repair capacities have been studied by Gross, Miller, and Soland (1981), but for steady state solutions only. This paper presents results on transient solutions for such problems.

For convenience in describing the various multi-echelon systems studied, the following classification scheme is adopted. A system is described by seven symbols, in the format $(n_1,n_2,n_3/n_4,n_5/n_6,n_7)$. Table I provides the definitions of n_1 through n_7.

As an example of the classification scheme, consider a military supply system with five bases, base and depot repair, base and depot

TABLE I

SYSTEM CLASSIFICATION SCHEME

Symbol	Definition
n_1	Number of bases (1,2,...)
n_2	Number of levels of repair (1,2,...)
n_3	Number of levels of supply (1,2,...)
n_4	Number of item types (1,2,...)
n_5	Number of levels of indenture (0,1,...)
n_6	Size of item population (f or ∞)
n_7	Capacity of repair facilities (f or ∞)

spares, two types of items with no indenture, finite calling population size (finite numbers of items), and finite repair capacities. This system would be represented as (5,2,2/2,0/f,f). The same system with infinite calling population and infinite repair capacity (Dyna-METRIC assumptions) would be denoted (5,2,2/2,0/∞,∞). The most general system descriptor would be (b,r,s/k,j/f,f); that is, a system with b bases, r levels of repair, s levels of supply, k item types, j levels of indenture, and finite calling populations and repair capacities. To describe fully any system, it would still be necessary actually to draw the network structure; nevertheless, this classification scheme will be most useful in delineating the systems under discussion.

Our goal is to present exact models of Markovian repairable item provisioning systems which can be solved numerically for exact transient state probabilities and other performance measures. We are also concerned with the practical problem of algorithmic implementation on a computer. It is possible to solve nontrivial, fairly complex, systems using the algorithmic approach of "randomization." We have developed highly efficient algorithms for two cases [(1,1,1/1,0/f,f) and (2,2,2/1,0/f,f)], and discuss generalizations of these situations.

The approach we use is a two stage procedure. The first stage is a method of modeling called *SERT* and the second is an algorithm based on the randomization technique [the reader is referred to Gross and Miller (1982) for a full methodological discussion of these]. A good discussion of the advantages of the randomization procedure as a numerical analysis tool is found in Grassmann (1977).

2. *SERT* MODELING APPROACH

Since we assume a finite Markovian system (all failure times and repair times are independent exponential random variables), the system can be fully described by its infinitesimal generator matrix

$$Q \equiv \lim_{\Delta t \to 0} \frac{[P(\Delta t)-I]}{\Delta t}$$

where $P(\Delta t) = \{p_{ij}(\Delta t)\}$; $p_{ij}(\Delta t) \equiv \Pr\{\text{system is in state } j \text{ at time } t + \Delta t \mid \text{in state } i \text{ at time } t\}$. The matrix Q is finite, say $N \times N$, since we are assuming a finite number of items in the system. For complex systems, N can be quite large (this will be seen later), so it is necessary to have algorithms as efficient as possible, both in running times and storage demands made on the computer.

It is not necessary to store this $N \times N$ Q matrix if we use the *SERT* modeling approach. Briefly [for more detail see Gross and Miller, *op. cit.*], we must describe the State space, the types of Events, the transition Rates (the off-diagonal nonzero elements of the Q matrix), and the Target states, that is, the state to which the system goes when a given type of event occurs.

Given a state space $\mathcal{S}$ of size N with s denoting a given state of the system ($s \in \mathcal{S}$), and an event space $\mathcal{E}$ of size E with e_j denoting an event of type j ($e_j \in \mathcal{E}$), it is necessary to consider a rate vector and a target state vector for each e_j , which we denote by $\underset{\sim}{r}^j$ and $\underset{\sim}{t}^j$, respectively. The dimensions of $\underset{\sim}{r}^j$ and $\underset{\sim}{t}^j$ are $1 \times N$. Since in these models $E << N$, it is much more efficient with respect to computer storage requirements to work only with the $\underset{\sim}{r}^j$ and $\underset{\sim}{t}^j$ vectors, rather than the $N \times N$ Q matrix, as the former requires consideration of only $2 \cdot E \cdot N$ elements rather than N^2 .

The *SERT* modeling procedure will be illustrated on the (1,1,1/1,0/f,f) and (2,2,2/1,0/f,f) systems to follow.

3. RANDOMIZATION COMPUTATIONAL TECHNIQUE

Transient solutions to Markovian queues require solving the set of differential equations

$$\underset{\smile}{\pi}'(t) = \underset{\smile}{\pi}(t)Q$$

where $\underset{\smile}{\pi}(t)$ is the vector of transient probabilities, i.e., $\underset{\smile}{\pi}(t) = \{\pi_j(t)\}$, where $\pi_j(t) \equiv \Pr\{\text{system is in state } j \text{ at time } t\}$ and $\underset{\smile}{\pi}'(t)$ is the vector of derivatives with respect to time, i.e., $\underset{\smile}{\pi}'(t) = \{\pi_j'(t)\}$. Many numerical techniques can be used to solve these linear, first order differential equations; for example, numerical integration [see Maron (1982)]. We choose instead a technique referred to as randomization [see Gross and Miller, *op. cit.*, or Grassmann (1977)], which is ideally suited to the *SERT* modeling of Markovian systems.

Randomization is based upon the ability to transform the continuous parameter Markov process analysis to an analysis of a discrete parameter Markov chain (MC) whose transition times are generated by a Poisson process.

Consider the continuous parameter process we are modeling. It remains in a given state s until one of many (E possible) events occurs, which then changes its state to, say, s' (if it is an event of type e_j, $s' = t_s^j$). Since all times are exponential, the time to the next event is the minimum of exponentials which is also exponential. Hence, if we look at the process only at transition times, it is a discrete parameter Markov chain, whose holding (transition) times are exponentially distributed.

Since the mean of the holding time in state s may depend on s, we do not quite have a Poisson process generating these transitions. To get around this, we denote the mean of the *minimum* holding time over all states (this always exists since S is finite) as $1/\Lambda$ and consider a Poisson process with rate Λ as a generator of the transitions. Because this generates transitions at a greater rate than desired (since Λ is the *maximum* of the state-dependent *rates*), we must *thin* the process to model the actual state-dependent holding times by adjusting the discrete parameter MC transition probability matrix. Without the adjustment, the embedded discrete parameter MC has transition probability matrix $P = \{p_{ij}\}$, $p_{ij} = q_{ij} / \sum_{j \neq i} q_{ij}$. The parameter Λ is $\max_i \sum_{j \neq i} q_{ij}$. Adjusting the P matrix for the thinning operation gives the transition probability matrix $\tilde{P} = \{\tilde{p}_{ij}\}$, where

$$\tilde{p}_{ij} = \frac{q_{ij}}{\sum_{j \neq i} q_{ij}} \cdot \frac{\sum_{j \neq i} q_{ij}}{\Lambda} = \frac{q_{ij}}{\Lambda} .$$

Thus, the computing formula for $\underset{\sim}{\pi}(t) = \{\pi_s(t)\}$ is

$$\pi_s(t) = \sum_{n=0}^{\infty} \sum_{i=0}^{N} \phi_s(n) \frac{e^{-\Lambda t}(\Lambda t)^n}{n!} , \tag{1}$$

where $\phi_s(n)$ is the probability that the system is in state s after n transitions generated by the Poisson (Λ) process, and $e^{-\Lambda t}(\Lambda t)^n/n!$ is the probability that there are n transitions of the Poisson (Λ) process in time $(0,t)$. The $\phi_s(n)$ can be determined using $\tilde{P}$ and $\pi_i(0)$ in the standard MC way; that is,

$$\begin{aligned} \underset{\sim}{\phi}(n) &= \underset{\sim}{\pi}(0)(\tilde{P})^n \\ &= \phi(n-1)\tilde{P} , \end{aligned} \tag{2}$$

where $\underset{\sim}{\pi}(0)$ is the starting state probability vector. The infinite sum in (1) must be truncated at some appropriate point (see Gross and Miller, *op. cit.*) which can be set to guarantee a bound on the error.

Using the target and rate vectors from the *SERT* procedure, we can compute the $\underset{\sim}{\phi}(n)$ vectors recursively in a manner more efficient than using (2) by the following algorithm:

(i) $\underset{\sim}{\phi}(0) = \pi(0)$

(ii) $\underset{\sim}{\phi}(n+1)$ is computed from $\underset{\sim}{\phi}(n)$ as *follows:*

(a) $$\phi_s(n+1) = \phi_s(n) \cdot \left(1 - \frac{\sum_{j=1}^{E} r_s^j}{\Lambda}\right) \qquad (3)$$

then

(b) for $j = 1,2,\ldots,E$, and $s \in S$, add

$\phi_s(n) \cdot (r_s^j/\Lambda)$ to $\phi_{t_s^j}(n+1)$.

What this algorithm does is to operate simultaneously on all components of $\underset{\sim}{\phi}(n+1)$ using the components of $\underset{\sim}{\phi}(n)$ and the transition probabilities from $\tilde{P}$. Note that (r_s^j/Λ) is $\tilde{p}_{s,t_s^j}$; that is, the probability of going from state s to state t_s^j (the target state that event j causes the system to switch to when it is in state s) given an occurrence of the Poisson (Λ) process. The term $(1 - \sum_{j=1}^{E} r_s^j / \Lambda)$ is the probability of ignoring an occurrence of the Poisson (Λ) process, which is called thinning and which leaves the system in state j (a null event).

This algorithm is very efficient for sparse Q matrices, which is the situation we generally have for repairable item systems, since it avoids the zero multiplication that would result from using the matrix multiplication of (2).

4. SIMPLE NETWORK MODEL: (1,1,1/1,0/f,f) SYSTEM

The simplest case of a repairable item system is the simple machine repair model shown in Figure 1. The situation modeled has a population consisting of M items desired to be operational at all times and Y spares to support the system. There are c repair channels, so that a maximum of c items can be undergoing repair simultaneously. If more than c items require repair, a queue forms at the repair facility. Failure and repair times are exponentially distributed random variables with the mean time to failure of any item denoted by $1/\lambda$ and the mean time to repair an item denoted by $1/\mu$; i.e., λ and μ are the failure rate and repair rate, respectively. The total number of items in the system is $M + Y \equiv N$ and if s items are in repair, then $N - s$ items are at the operating node, so that when $s > Y$, the population is operating at less than the desired strength of M .

For modeling purposes, the operating node of the two-node network pictured in Figure 1 can be considered to be an M channel queue, so

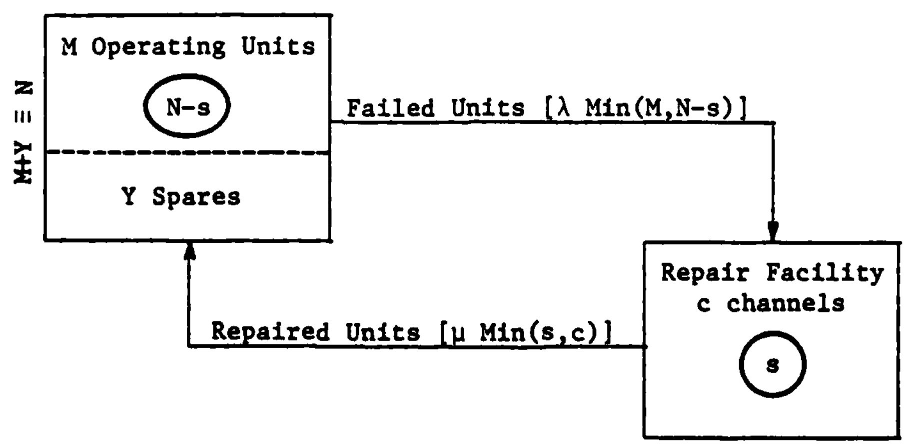


Figure 1.--(1,1,1/1,0/f,f) system.

that if there are more than M units at this node ($s < Y$), the queue represents on-hand spares. If no queue is present and some "servers" are idle at the operating node ($s > Y$), then the population is operating at degraded strength (as mentioned above).

Utilizing the *SERT* methodology, the state of the system s can be considered to be the number of items in or awaiting repair (in resupply). The size of S is $N = M + Y + 1$, since $s = 0,1,2,\ldots,M + Y$.

There are only two types of events ($E = 2$). The first type of event is a failure. The rate vector for failures is $\underset{\smile}{r}^f = (r_0^f, r_1^f, \ldots, r_s^f, \ldots, r_N^f)$, where

$$r_s^f = \begin{cases} M\lambda & 0 \leqslant s \leqslant Y \\ (M + Y - s)\lambda & Y \leqslant s \leqslant N \end{cases}$$

and the target state vector is $\underset{\smile}{t}^f = (t_0^f, t_1^f, \ldots, t_s^f, \ldots, t_N^f)$, where

$$t_s^f = \begin{cases} s + 1 & s = 0,1,\ldots,N - 1 \\ N & s = N \end{cases}$$

since a failure increases the number in resupply by one, except when N are in resupply. In this case a failure cannot occur, but we show t_N^f as N; that is, a "mythical" failure does not change the state.

The second type of event is a service completion. The sth component of the rate vector $\underset{\smile}{r}^r$ is

$$r_s^r = \begin{cases} s\mu & 0 \leqslant s \leqslant c \\ c\mu & c \leqslant s \leqslant N \end{cases}$$

and the corresponding sth component of the $\underset{\smile}{t}^r$ vector is

$$t_s^r = \begin{cases} s - 1 & 1 \leqslant s \leqslant N \\ 0 & s = 0 . \end{cases}$$

We can now apply the *SERT* multiplication algorithm (3), which reduces to

(i) $\underset{\smile}{\phi}(0) = \pi(0)$

(ii) (a) For $s = 0,1,\ldots,N$, set

$$\phi_s(n+1) = \phi_s(n) \cdot \left(1 - \frac{r_s^f + r_s^r}{\Lambda}\right) ,$$

then

(b_1) For $s = 0,1,\ldots,N$, add

$$\phi_s(n) \cdot \frac{r_s^f}{\Lambda} \text{ to } \phi_{t_s^f}(n+1)$$

(b_2) For $s = 0,1,\ldots,N$, add

$$\phi_s(n) \cdot \frac{r_s^r}{\Lambda} \text{ to } \phi_{t_s^r}(n+1) .$$

From $\underset{\smile}{\phi}(n)$, we can obtain $\underset{\smile}{\pi}(t)$ rather easily using equation (1), and use $\underset{\smile}{\pi}(t)$ to obtain system performance measures, such as system availability, which we define as the probability that the desired number of components (M) are operating at time t ; that is,

$$A(t) = \sum_{s=0}^{Y} \pi_s(t)$$

Various other measures such as expected backorder level, expected number of units operating, etc., can also be readily calculated from $\underset{\smile}{\pi}(t)$.

Computational efficiency for system performance measures can be gained by using $\underset{\smile}{\phi}(n)$ and then converting to continuous time (see Gross and Miller, *op. cit.*), for example, defining

$$A_n = \sum_{s=0}^{Y} \phi_s(n) ,$$

we have

$$A(t) = \sum_{n=0}^{T} A_n \frac{e^{-\Lambda t}(\Lambda t)^n}{n!} .$$

This algorithm has been coded as an interactive FORTRAN program called REPTRAN1. Sample input and output [A(t) vs. t] are shown in Figures 2 and 3, respectively. Also shown is the equivalent (1,1,1/1,0/∞,∞) calculations--the Dyna-METRIC model. This has also been coded up as part of the REPTRAN1 program.

Finite source, finite repair models (-,-,-/-,-/f,f) are referred to as "closed queuing network" models, for these types of models can indeed be viewed as closed queuing networks [the (1,1,1/1,0/∞,∞) of REPTRAN1 is the simplest closed network, namely, a two-stage cyclic queue].

5. SPECIAL CASE: (2,2,2/1,0/f,f)

We now present an implementation of the techniques discussed previously to the computation of transient probabilities and availabilities of a (2,2,2/1,0/f,f) system. (It has been coded as a FORTRAN program called REPTRAN2.) The system is shown in Figure 4. Items of one type move around a network with six nodes, namely, operational at Base 1 (BU1), in or awaiting repair at Base 1 (BR1), operational at Base 2 (BU2), in or awaiting repair at Base 2 (BR2), operational (ready spares) at depot (DU), and in or awaiting repair at depot (DR). The description of the system will follow the approach used earlier. The number of operating machines

```
$ RUN REPTRAN1

THIS IS REPTRAN1 PROGRAM

DO YOU WANT A HARDCOPY (Y OR N) ?
N
DO YOU WANT TO PRINT ALL OUTPUT (Y OR N) ?
Y
Type:  Initial time (assume zero),final time, time increment
0,15,1
NUMBER OF TIME POINTS =  16       NUMBER OF TIME PERIODS =  15
Case number:     1
DO YOU WANT TO RUN A CLOSED QUEUEING NETWORK OR  A DYNAMETRIC MODEL
TYPE 1 OR 2 ACCORDINGLY
1
Type: M,C,Y,EPSILON
3,2,2,.001
Type number of lambda values to be used
2
Type lambda values
.2,.3
Type times of shift of lambdas (start w/ 0)
0,6
Type number of mu values to be used
2
Type mu values
.5,.75
Type times of shift of mu (start w/0)
0,10
Do you want to type in an initial prob. vector ? (Type Y or N)
If you type N, program assumes (1,0,0,...,0) as
initial prob. vector.
N


DO YOU WANT TO RUN ANOTHER CASE?
Y
Case number:     2
DO YOU WANT TO RUN A CLOSED QUEUEING NETWORK OR  A DYNAMETRIC MODEL
TYPE 1 OR 2 ACCORDINGLY
2
TYPE LAMBDA1,LAMBDA2 AND THE TIME AT WHICH  THE CHANGE OF LAMBDA OCCURS
.6,.9,6
TYPE MU1,MU2,MU3
.5,.5,.75
TYPE THE TIMES WHERE THE MUS CHANGE VALUES
```

Figure 2.--Sample input for REPTRAN1: Closed queuing network and Dyna-METRIC runs.

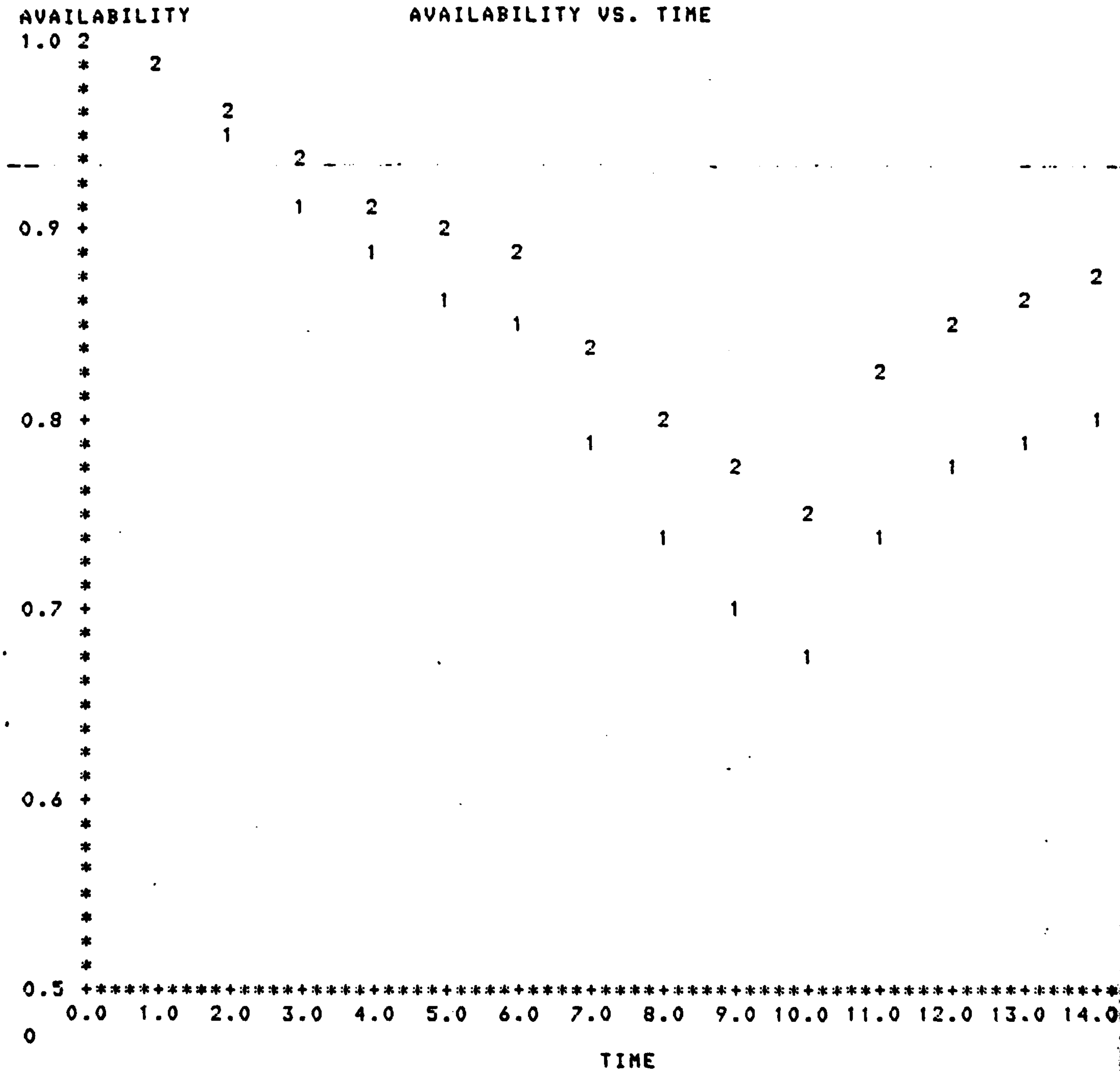


Figure 3.--Sample output for REPTRAN1: Closed queuing network (1) and Dyna-METRIC (2) runs.

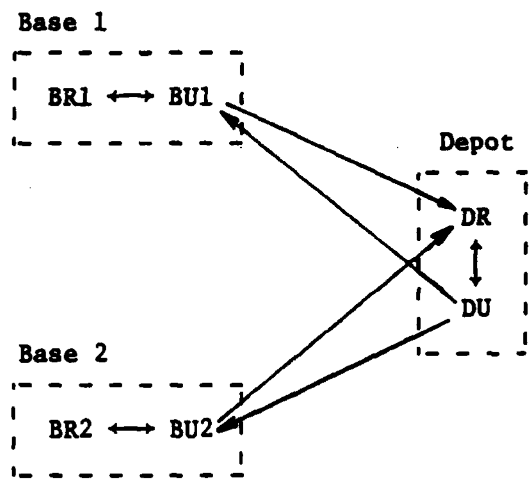


Figure 4.--General schematic for a (2,2,2/1,0/f,f) repairable item system.

at Base 1 equals the maximum of the desired number, MS1, and the number of machines at node BU1 (#BU1); similarly for Base 2.* The number of spares available at the depot spares pool is the number at node DU (#DU). The number of busy repair channels at Base 1 equals the minimum of the number of repair channels, BC1, and the number of items at node BR1 (#BR1); similarly for Base 2 and the depot. Thus, the system parameters (failure and repair rates) and the number of machines at each node give us total information about the system. Knowledge of how machines move around the network completes the description of the system. The only point which must yet be specified in detail is the assignment rule for filling backorders to the depot.

Consider the situation where the number of machines at node DR (#DR) is greater than the spares allocated to the depot, DS. In this case there will be no machines at node DU and a backorder level of #DR - DS at the depot.

*Quantities with "#" preceding them are system state variables; quantities without # are either node designators, event descriptors, or preset parameter values.

The question arises concerning the allocation of the next machine to complete repair at the depot: To which base should it be given? We must define an allocation function based on the state of the system. For now, let us define the state of the system as the number of machines at each node*:

$$\underline{s} = (\#BU1, \#BR1, \#BU2, \#BR2, \#DR, \#DU) .$$

The number of backorders at the depot from Bases 1 and 2, respectively, are

$$\#BD1 = BS1 - (\#BU1 + \#BR1)$$

$$\#BD2 = BS2 - (\#BU2 + \#BR2)$$

where BS1 and BS2 are the numbers of spares allocated to Bases 1 and 2, respectively. Possible allocation functions are

$$ALL1(s) = \begin{cases} 1 & \text{if } \#BD1 \geq \#BD2 \\ 2 & \text{if } \#BD2 > \#BD1 \end{cases}$$

or

$$ALL2(s) = \begin{cases} 1 & \text{if } \#BD1/BS1 \geq \#BD2/BS2 \\ 2 & \text{if } \#BD1/BS1 > \#BD2/BS2 \end{cases}$$

In the program REPTRAN2 we use a *randomized* generalization of the second allocation function: Let PROB.ALL(•) be a function that equals the probability of assigning the repaired machine to Base 1. The assignment rule is given in Table II. The weights W1 and W2 are supplied by the user and reflect his

TABLE II

ASSIGNMENT RULE FOR SENDING A REPAIRED MACHINE FROM DEPOT IN BACKORDER SITUATION

Condition	Probability of Assignment to Base 1	Probability of Assignment to Base 2
W1 * #BD1 > W2 * #BD2	1	0
W1 * #BD1 = W2 * #BD2	1/2	1/2
W1 * #BD1 < W2 * #BD2	0	1

*Because it is a closed system, we shall see later that $\underline{s}$ can be characterized by fewer state variables.

strategy for favoring one base over the other when both have backordered units.

The description of the system is now complete and we proceed to describe the Markov model of the system using the SERT approach. This requires (i) identifying the state space (as a vector), (ii) defining the event set, and for each event (iii) computing the vector of transition rates, and (iv) computing the vector of target states. The randomization algorithm can then be used to compute transient probabilities to any user-specified accuracy (ϵ). The events are given in Table III; once the state space is described as a one-dimensional vector, steps (iii) and (iv) are straightforward.

In general, the state space appears to have six dimensions, but because of one-for-one ordering and conservation of the total number of items in the system, the state space actually has a lower dimension.

TABLE III

THE EVENTS THAT ACCOUNT FOR ALL THE STATE CHANGES OF A (2,2,2/1,0/f,f) SYSTEM WITH A SINGLE TYPE OF ITEM

Name	Description
B1R	Repair completed at Base 1
B2R	Repair completed at Base 2
F1B	Failure at Base 1 (base repairable)
F2B	Failure at Base 2 (base repairable)
F1D	Failure at Base 1 (depot repairable)
F2D	Failure at Base 2 (depot repairable)
DR1	Repair completed at depot and sent to depot spares pool if not backorder situation; otherwise sent to Base 1
DR2	Repair completed at depot while backorder situation and sent to Base 2

The description of the state space breaks into two situations: no depot spares available, and some depot spares available,

$$S = S_0 \cup S_+$$

where

$$S_0 = \text{states with depleted depot spares pool}$$

$$S_+ = \text{states with nondepleted depot spares pool.}$$

First consider S_0 . In this case, it is possible to describe the state of the system with four numbers:

$$(\#BU1,\ \#BD1,\ \#BU2,\ \#BD2)$$

It is known that $\#DU = 0$ and the remaining machines are at node DR. The feasible states of S_0 are subject to two constraints:

$$\#BU1 + \#BD1 \leq BS1$$

$$\#BU2 + \#BD2 \leq BS2$$

and thus S_0 is a Cartesian product,

$$S_0 = T_1 \times T_2$$

where

$$T_1 = \{(\#BD1,\ \#BU1):\ \ \#BD1 + \#BU1 \leq BS1\}$$

$$T_2 = \{(\#BD2,\ \#BU2):\ \ \#BD2 + \#BU2 \leq BS2\}$$

These sets are shown in Figure 5. (The notation T is used because the spaces are triangular.) Note that the number of points in T_1 and T_2 are

$$|T_1| = \frac{(BS1+1)(BS1+2)}{2}$$

and

$$|T_2| = \frac{(BS2+1)(BS2+2)}{2}$$

respectively, and the number of states in S_0 is the product

$$|S_0| = \frac{(BS1+1)(BS1+2)(BS2+1)(BS2+2)}{4}$$

		#BU1					
		0	1	2	3	4	
	0	•	•	•	•	•	
	1	•	•	•	•		
#BD1	2	•	•	•			T_1
	3	•	•				
	4	•					

		#BU2						
		0	1	2	3	4	5	
	0	•	•	•	•	•	•	
	1	•	•	•	•	•		
	2	•	•	•	•			
#BD2	3	•	•	•				T_2
	4	•	•					
	5	•						

Figure 5.--Examples of state space to describe individual bases; in the case BS1 = 4 and BS2 = 5.

Now let us consider the states where the spares pool at the depot is not empty, S_+ . In this case the state of the system can be described by three numbers:

$$(\#BU1, \#BU2, \#DU)$$

The constraints on these are

$$\#BU1 \le BS1$$

$$\#BU2 \le BS2$$

$$1 \le \#DU \le DS$$

We can condition on the value of #DU to get S_+ into the form

$$S_+ \cup S_1 \cup S_2 \cup \ldots \cup S_{DS}$$

where S_i consists of states with exactly i machines in the depot spares *pool*; such a set is depicted in Figure 6. Note that each S_i is a rectangle and

$$|S_i| = (BS1 + 1)(BS2 + 1)$$

Thus

$$|S_+| = (BS1 + 1)(BS2 + 1)DS$$

and the total number of states is

$$|S| = \frac{(BS1+1)(BS1+2)(BS2+1)(BS2+2)}{4} + (BS1 + 1)(BS2 + 1)DS$$

Examples of state space sizes are given in Table IV.

Although in principle we can apply the randomization algorithm to a system whose state space is described in this complex multidimensional form, we choose to work with one-dimensional state spaces. We made this choice for two reasons. First, one general implementation of the randomization algorithm will work on all systems after they are put into

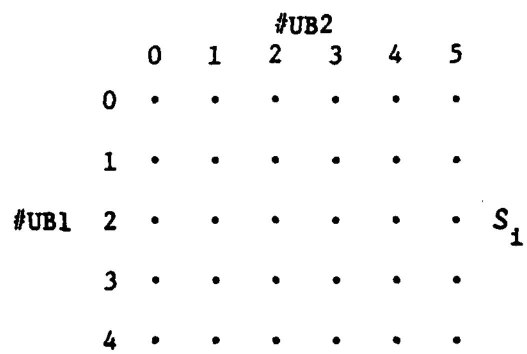


Figure 6.--The state space describing status of bases given that depot spares are available; BS1 = 4 and BS2 = 5.

TABLE IV

SIZE OF THE STATE SPACES OF (2,2,2/1,0/f,f) SYSTEM FOR SELECTED VALUES OF BS1, BS2, AND DS

BS1	BS2	DS	$\lvert S_0 \rvert$	+ $\lvert S_+ \rvert$	= $\lvert S \rvert$
2	2	2	36	18	54
4	4	2	225	50	275
6	6	2	784	98	882
8	8	2	2025	162	2187
10	10	2	4356	242	4598
12	12	2	8281	338	8619
18	18	2	36100	722	36822
24	24	2	105625	1250	106875

one-dimensional form, and it seems to decrease the problems in verifying the program. Second, and even more important, working directly with vectors rather than structures such as

$$S = T_1 \times T_2 \cup \bigcup_{i=1}^{DS} S_i$$

will speed up the algorithm by a significant factor because it is much faster to "fetch" and "store" an element of a one-dimensional vector than it is to "fetch" and "store" an element of a complicated multidimensional array. We estimate that this speeds up REPTRAN2 by a factor of four or more.

Thus, let us consider putting the elements of

$$S = T_1 \times T_2 \cup \bigcup_{i=1}^{DS} S_i$$

into a one-dimensional form. First consider T_1; this triangular shaped region can be transformed into a linear space L_1 by placing the rows end to end,

$$L_1 = \{00, 01, 02, 03, 04, 10, 11, 12, 13, 20, 21, 22, 30, 31, 40\}$$

Doing the same thing for T_2 yields L_2; thus,

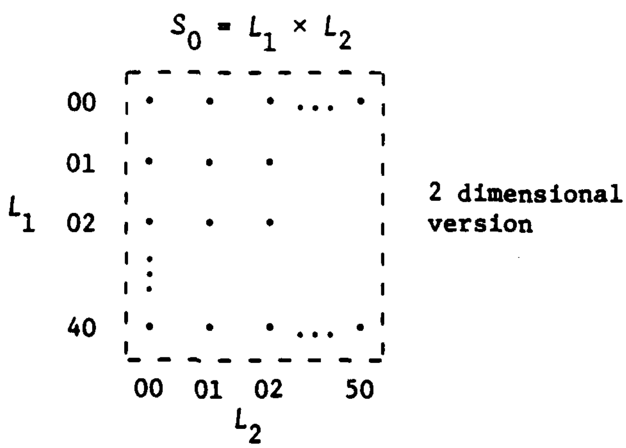
$$S_0 = T_1 \times T_2 \equiv L_1 \times L_2$$

becomes a two-dimensional set which can be linearized again by placing the rows end to end as shown graphically in Figure 7. The first $|S_0|$ elements of our one-dimensional vector will correspond to the states in S_0 in linear order as presented in Table IV.

The remaining $|S_+|$ elements in the one-dimensional vector will correspond to the states in S_+ arranged linearly as follows:

$$S_+ = \bigcup_{i=1}^{DS} S_i ,$$

where S_i is depicted in Figure 6. Each S_i is put into one-dimensional form by putting rows end to end (as in Figure 7). Finally, these DS



0000, 0001, ..., 0050, 0100, 0101, ..., 0150, ..., 4000, 4001, ..., 4050

Linear version

Figure 7.--Depiction of linearization of the Cartesian product being transformed into a one-dimensional space.

one-dimensional S_i's are put end to end starting with S_1 and ending with S_{DS}. (Thus the last state in this one-dimensional listing is (BS1, BS2, DS), which corresponds to BS1 units "up" at Base 1 (#BU1 = BS1), BS2 units "up" at Base 2, and DS units in the spares pool at the depot, i.e., a perfect system with no failed units. The first state listed in the one-dimensional version of the state space is (0, 0, 0, 0), which corresponds to BS1 units in repair at Base 1, BS2 units in repair at Base 2, and DS units in repair at the depot.)

This completes the description of the state space of this (2,2,2/1,0/f,f) system as a one-dimensional vector. It is used by the program REPTRAN2. A major part of REPTRAN2 is subroutines that take the parameters and rates and compute one-dimensional vectors of transition rates and one-dimensional vectors of target states for each event in Table III. It then uses the randomization algorithm logic to compute transient state probabilities and availabilities as described in Section 2.

The program REPTRAN2 can also compute transient probabilities for (2,2,2/1,0/f,f) systems whose underlying rates (failure and repair rates) change at discrete points in time (on a lattice time scale)--up to five changes are accommodated for each underlying rate.

We note that by setting certain parameters to special values, REPTRAN2 can handle the corresponding special cases. If DS = 0, we lose the second level of supply and get a (2,2,1/1,0/f,f) system. If FB1 = 1 then all failures at Base 1 are base repairable and Base 1 does not interact with the depot: thus Base 1 is a (1,1,1/1,0/f,f) system and Base 2 and the depot form a (1,2,2/1,0/f,f) system, or (1,2,1/1,0/f,f) if DS = 0.

6. COMPUTER RUNS

The REPTRAN2 program has been run for various (2,2,2/1,0/f,f) systems as described in Section 5. The program executes in quite short times for systems with large state spaces. It also compares the probabilities computed by the Dyna-METRIC model for corresponding (-,-,-/-,-/∞,∞) systems, which are also a part of the REPTRAN2 code. [For the Dyna-METRIC developments, details for these corresponding systems, see Gross, Kioussis, and Miller (1982).] Furthermore, it is reassuring that when REPTRAN2 was run on a system whose parameters gave rise to a (1,1,1/1,0/f,f) subsystem (as discussed in Section 4), the program gave the same answers as REPTRAN1.

REPTRAN2 is interactive and the user has the option of running either the Dyna-METRIC model or the closed queuing network model for a (2,2,2/1,0/•,•) system. Availability is defined as the probability that the number of machines "up" at a base meets or exceeds the number desired to be operating, and is the primary measure outputted.

We ran a few sample cases to get an idea of how well this program performs (with reference to time) in computing exact transient solutions. The sample cases we ran are described in Table V. For each case, all the necessary parameters (which the user must supply) are listed. The failure rates and repair rates shift at the times indicated for each case.

For example, in Case 1a: At time $t_0 = 0$, the failure rate of a machine at Base 1 is $\lambda_1 = .4$; then at time $t_{\lambda_1} = 6$, this rate changes to $\lambda_1 = .6$. At time $t_0 = 0$ the repair rate of a single repair channel at Base 1 is $\mu_1 = .5$; then at time $t_{\mu_1} = 10$, this rate changes to $\mu_1 = .75$. Similar changes occur for the failure and repair rates at Base 2 and the repair rate at the depot.

TABLE V

PARAMETER VALUES OF THE SAMPLE CASES SOLVED USING REPTRAN2

A. Input Parameter Names	
Symbol	Meaning
BSi	Allocation of total stock to Base i (operating machines plus spares), i = 1,2
MSi	Desired number of working machines at Base i
BCi	Number of repair channels in repair shop at Base i
FBi	Proportion of items failing at Base i, base repairable
t_0	Time zero
t_{λ_i}	Time of shift in mean failure rate, Base i
t_{μ_i}	Time of shift in mean repair rate, Base i
$\lambda_i(t_0)$	Initial mean failure rate, Base i items
$\lambda_i(t_{\lambda_i})$	Shifted mean failure rate, Base i items
$\mu_i(t_0)$	Initial mean repair rate, Base i items
$\mu_i(t_{\mu_i})$	Shifted mean repair rate, Base i items
Wi	Weighting factor for filling depot backorders for Base i
DS	Number of depot spares
DC	Number of depot repair channels
$\mu_D(t_0)$	Initial mean depot repair rate
t_D	Time of shift in mean depot repair rate
$\mu_D(t_D)$	Shifted mean depot repair rate
ε	Error tolerance

Table V--*continued*

B. REPTRAN2 Run Description

BS1	MS1	BC1	FB1	$\lambda_1(t_0)$	$\lambda_1(t_{\lambda_1})$	$\mu_1(t_0)$	$\mu_1(t_{\mu_1})$
BS2	MS2	BC2	FB2	$\lambda_2(t_0)$	$\lambda_2(t_{\lambda_2})$	$\mu_2(t_0)$	$\mu_2(t_{\mu_2})$
DS	DC	W1	W2	$\mu_D(t_0)$	$\mu_D(t_D)$		
ε							
				Cases			
1a							
4	2	2	.7	.4(0)	.6(6)	.5(0)	.75(10)
5	3	2	.5	.4(0)	.6(8)	.6(0)	.9(12)
2	2	.4	.6	.3(0)	.45(11)		
.001							
1b							
4	2	2	.7	.4(0)	.6(6)	.5(0)	.75(10)
5	3	2	.5	.4(0)	.6(8)	.6(0)	.9(12)
2	2	.5	.5	.3(0)	.45(11)		
.001							
2a							
4	2	2	.7	.2(0)	.4(6)	.5(0)	.75(10)
5	3	2	.5	.1(0)	.15(8)	.4(0)	.6(12)
2	2	.4	.6	.3(0)	.45(11)		
.001							

Table V--*continued*

2b							
4	2	3	.7	.2(0)	.3(6)	.5(0)	.75(10)
5	3	3	.5	.1(0)	.15(8)	.4(0)	.6(12)
2	3	.4	.6	.3(0)	.45(11)		
.001							
3							
4	2	2	.7	.2(0)	.3(6)	.5(0)	.75(10)
5	3	2	.5	.1(0)	.15(8)	.4(0)	.6(12)
2	2	.4	.6	.3(0)	.45(11)		
.00001							
4							
8	4	4	.7	.4(0)	.6(6)	.5(0)	.75(10)
10	6	4	.5	.4(0)	.6(8)	.6(0)	.9(12)
4	4	.5	.5	.3(0)	.45(11)		
.001							
5							
18	14	2	.6667	.2(0)	.3(6)	1.(0)	1.5(10)
13	10	2	.6667	.143(0)	.2143(6)	1.(6)	1.5(10)
3	4	.5	.5(0)	.75(10)			
.0001							

The seven cases in Table V were solved exactly over 16 time points {0,1,...,15}. The CPU execution times are shown in Table VI, as well as the size of the state space of the model. Also shown is the set of time points at which rate changes (failure or repair) occurred.

TABLE VI

SIZE OF STATE SPACE AND RUNNING TIMES FOR REPTRAN2 SOLUTION OF VARIOUS (2,2,2/1,0/f,f) SYSTEMS

System Number	Size of State Space	Homogeneous Time Interval	CPU (sec.)[a]
1a	375	0, 6, 8, 10, 11, 12	10.18
1b	375	0, 6, 8, 10, 11, 10	10.48
2a	375	0, 6, 8, 10, 11, 12	8.07
2b	375	0, 6, 8, 10, 11, 12	8.87
3	375	0, 6, 8, 10, 11, 12	9.04
4	3366	0, 6, 8, 10, 11, 12	190.25
5	20748	0, 6, 10	1393.73

[a]Running time is approximately proportional to size of state space × number of event types × number of occurrences of Poisson process.

The seven cases in Table VI were also solved using the Dyna-METRIC approximate model (2,2,2/1,0/∞,∞). For this comparison it was necessary to select failure arrival rates at each base. We chose an arrival rate of MS1 * λ_1 at the Base 1 repair shop and MS2 * λ_2 at the Base 2 repair shop. For systems operating at moderate to high availabilities, these should be approximately correct.

We show the computer input and output for case 5 given in Table V as Figures 8 through 12. The interactive input is shown, first modelled as an exact queuing network (2,2,2/1,0/f,f) and then modelled as a Dyna-METRIC model (2,2,2/1,0/∞,∞). Both models were solved for availabilities and then superimposed on three plots. The exact availabilities are

```
TYPE: INITIAL TIME (assume zero),FINAL TIME, TIME INCREMENT
0,15,1
NUMBER OF TIME POINTS =    16    NUMBER OF TIME PERIODS =  15

 CASE NUMBER :   1
DO YOU WANT TO RUN A CLOSED QUEUEING NETWORK OR   A DYNAMETRIC MODEL
TYPE 1 OR 2 ACCORDINGLY
1

TYPE BS1,MS1,BC1,AND FB1
18,14,2,.6667

BS1 :  18  MS1 :  14  BC1 :   2  FB1 :0.6667

TYPE BS2,MS2,BC2,AND FB2
13,10,2,.6667

BS2 :  13  MS2 :  10  BC2 :   2  FB2 :0.6667

TYPE DS,DC,W1,W2
3,4,.5,.5

DS :   3  DC :   4  W1 :0.5000  W2 :0.5000
 TYPE NUMBER OF FAILURE RATES TO BE USED IN BASE 1
2
 TYPE THE FAILURE RATES
.2,.3
 TYPE THE TIME WHERE THOSE CHANGES IN RATES OCCUR  (START WITH 0)
0,6
 TYPE NUMBER OF REPAIR RATES TO BE USED IN BASE 1
2
 TYPE THE REPAIR RATES
1.,1.5
 TYPE THE TIME WHERE THOSE CHANGES IN RATES OCCUR  (START WITH 0)
0,10
 TYPE NUMBER OF FAILURE RATES TO BE USED IN BASE 2
2
 TYPE THE FAILURE RATES(BASE 2)
.143,.2145
 TYPE THE TIME WHERE THOSE CHANGES IN RATES OCCUR  (START WITH 0)
0,6
 TYPE NUMBER OF REPAIR RATES TO BE USED IN BASE 2
2
 TYPE THE REPAIR RATES (BASE 2)
1.0,1.5
 TYPE THE TIME WHERE THOSE CHANGES IN RATES OCCUR  (START WITH 0)
0,10
 TYPE NUMBER OF REPAIR RATES TO BE USED IN DEPOT
2
 TYPE THE REPAIR RATES (DEPOT)
.5,.75
 TYPE THE TIME WHERE THOSE CHANGES IN RATES OCCUR  (START WITH 0)
0,10
 TYPE THE DESIRED MAGNITUDE OF THE ERROR(EPSILON)
.001
```

Figure 8.--Sample input for REPTRAN2 case 5, closed queuing network.

```
DO YOU WANT TO RUN ANOTHER CASE?(Y or N)
Y
 CASE NUMBER :  2
DO YOU WANT TO RUN A CLOSED QUEUEING NETWORK OR   A DYNAMETRIC MODEL
TYPE 1 OR 2 ACCORDINGLY
2
 TYPE THE TIME AT WHICH THE FAILURE RATE AND THE REPAIR RATE CHANGE AT BASE
6,10
 TYPE THE TWO FAILURE RATES FIRST AND THEN THE TWO REPAIR RATES
2.8,4.2,1.0,1.5
 TYPE THE STOCK LEVEL AT BASE 1 AND THE PROPORTION OF THE FAILED ITEMS GOI
BASE REPAIR 1
4,.6667
 TYPE THE TIME AT WHICH THE FAILURE RATE AND THE REPAIR RATE CHANGE AT BASE
6,10
 TYPE THE TWO FAILURE RATES FIRST AND THEN THE TWO REPAIR RATES (BASE 2)
1.43,2.143,1.0,1.5
 TYPE THE STOCK LEVEL AT BASE 2 AND THE PROPORTION OF THE FAILED ITEMS GOI
BASE REPAIR 2
3,.6667
 TYPE THE TIME AT WHICH THE REPAIR RATE CHANGES AT THE DEPOT
10
 TYPE THE TWO REPAIR RATES (FOR DEPOT)
.5,.75
 TYPE THE STOCK LEVEL AT THE DEPOT
3
```

Figure 9.--Sample input for REPTRAN2 case 5, Dyna-METRIC.

plotted using the symbol "1" and the approximate Dyna-METRIC availabilities are plotted using the symbol "2." For time points where the symbol "1" fails to appear, it coincides with the "2." "Availability 1" is availability at Base 1, "Availability 2" is availability at Base 2, and "Availability 3" is simultaneous availability at both Base 1 and Base 2.

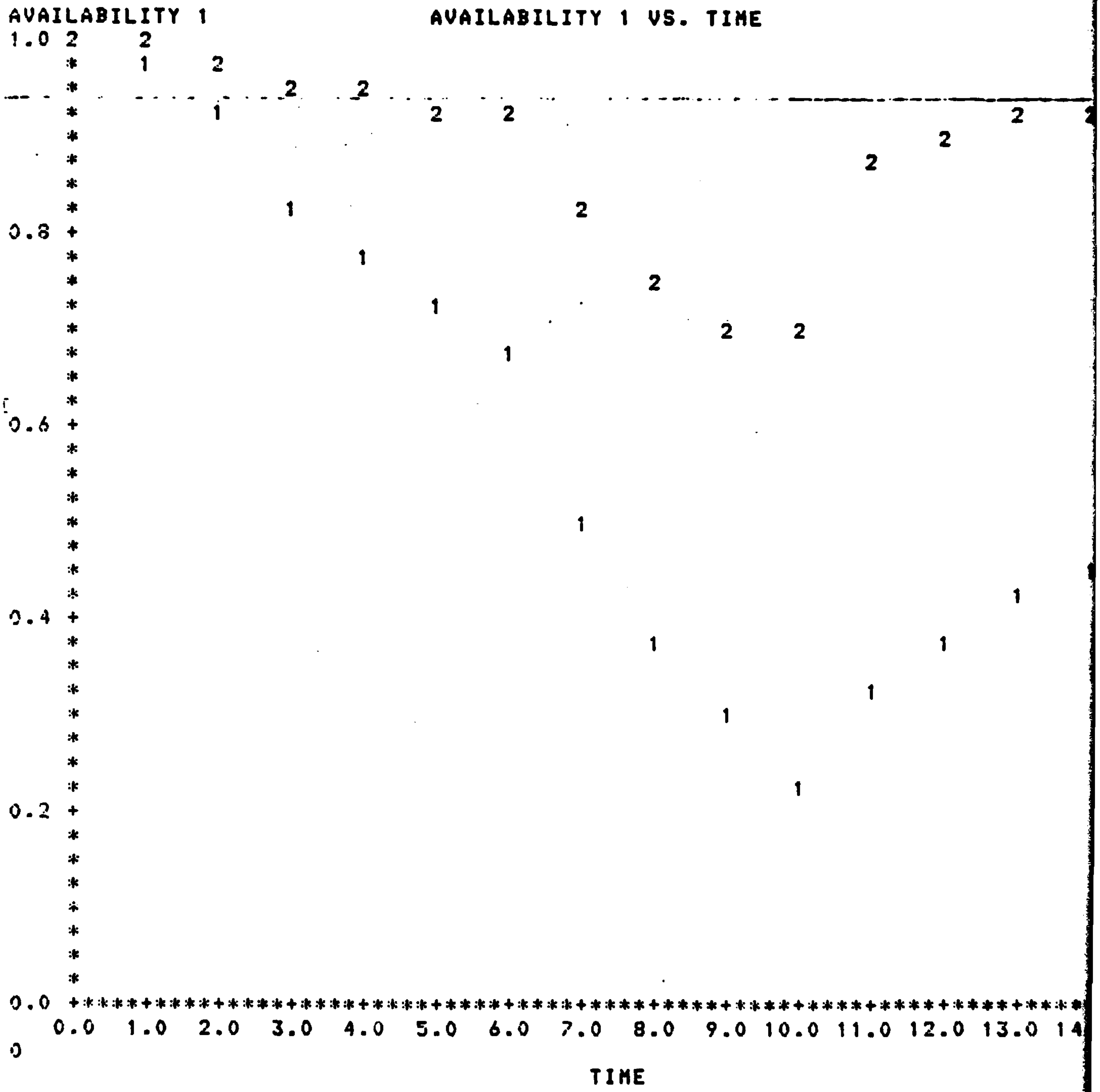


**Figure 10.--Sample output for REPTRAN2 case 5,
1: closed queuing network,
2: Dyna-METRIC,
availabilities at Base 1.**

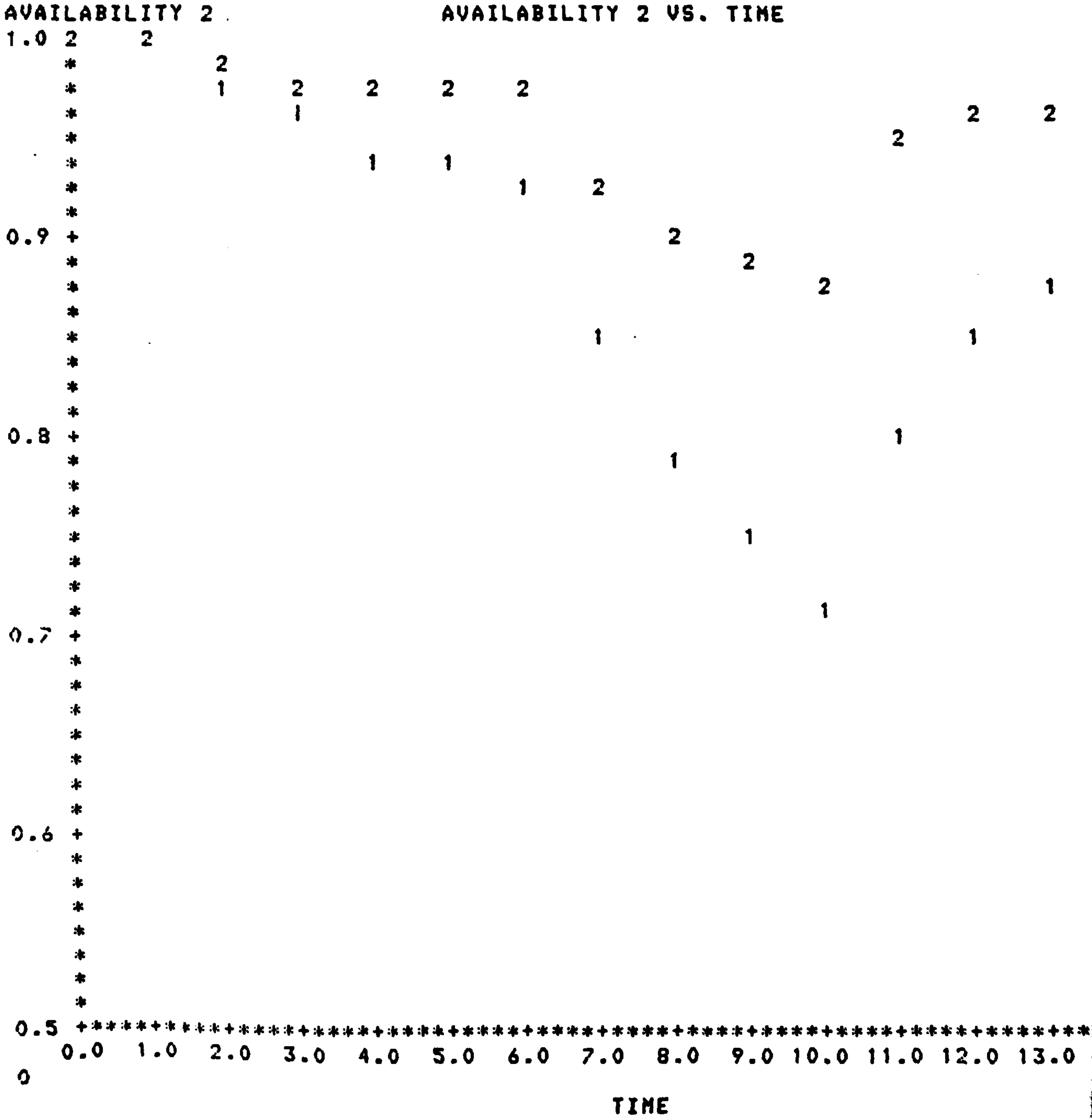


Figure 11.--Sample output for REPTRAN2 case 5,
1: closed queuing network,
2: Dyna-METRIC,
availabilities at Base 2.

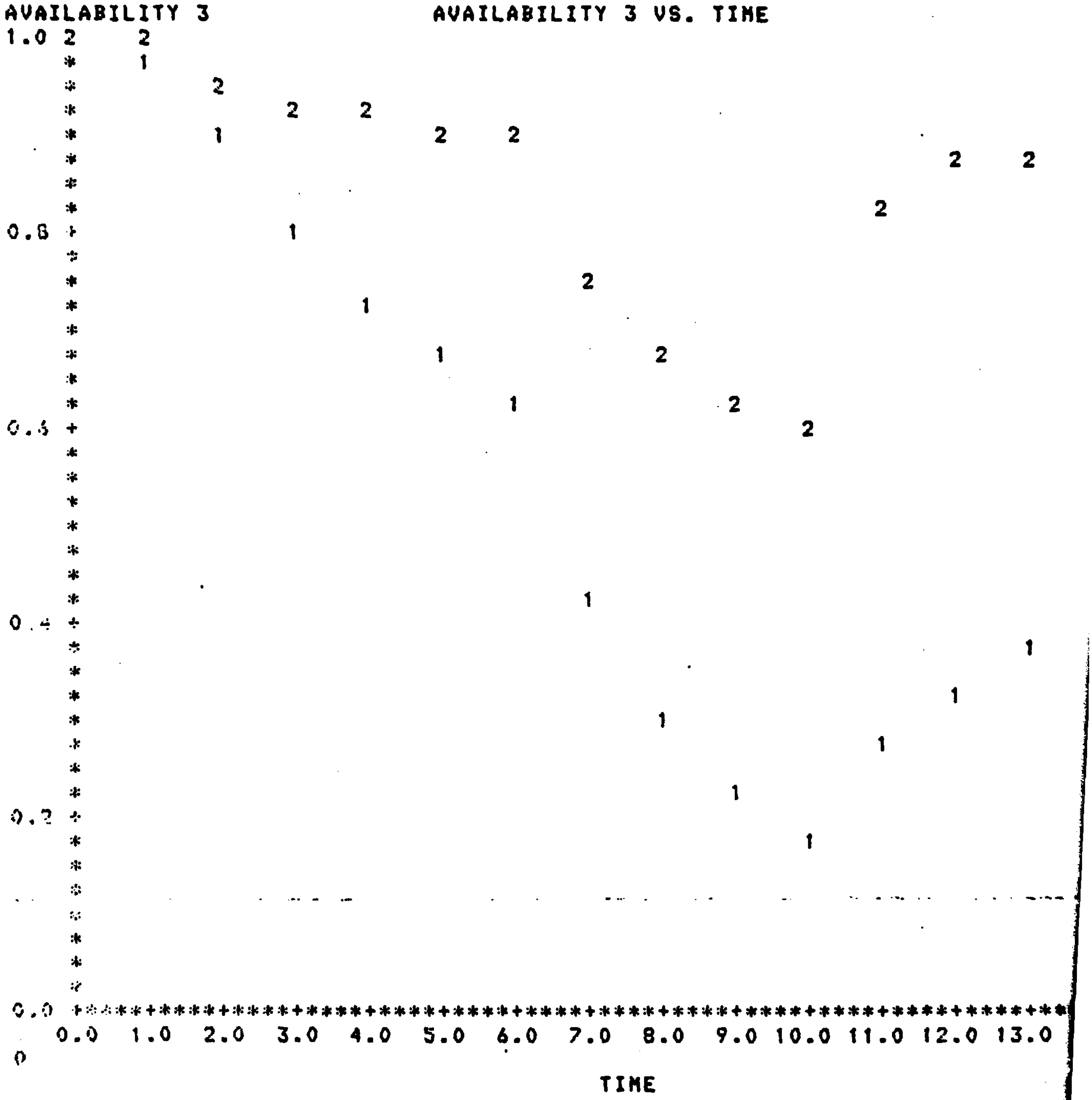


Figure 12.--Sample output for REPTRAN2 case 5,
1: closed queuing network,
2: Dyna-METRIC,
simultaneous availabilities at
both bases.

7. CONCLUSIONS

The *SERT* modelling technique and randomization computing algorithm [methodology developed in Gross and Miller (1982)] has been applied and implemented for computing transient performance measures of multi-echelon repairable item inventory systems. We have shown that it is feasible to compute exact probabilities for systems with large state spaces (20,000 or more states). Furthermore, for the types of systems under consideration, we believe that significantly larger cases are feasible using a truncated state space approach; that is, lumping the vast number of very low probability states together as one.

For example, in a (1,1,1/1,0/f,f) case with 31 states ($M = 20$, $Y = 10$), we found that the probabilities of s units in resupply for $s \geq 15$ was zero to at least three significance figures. Lumping states $s = 15, 16, \ldots, 31$ together would reduce the problem from 31 states to 16, a savings of almost 50%. We estimate in the 20,000 state space example that such a procedure would easily cut the number of states in half.

We have used the *SERT* modelling technique on a (b,2,2/k,1/f,f) system [see Gross, Kioussis, and Miller (1982)]. This has not been coded, but for moderate b and k, using the truncated state space approach, development of an efficient code should be feasible. Conceptually, of course, the most general (b,r,s/k,j/f,f) system could be modelled using *SERT*; the problem, of course, is the state space size for cases other than those with very small values of b, r, s, k, and j. The truncated state space approach offers the most promise for treating these models.

ACKNOWLEDGMENT

The authors thank Mr. L. Kioussis for his outstanding programming efforts, that resulted in the REPTRAN1 and REPTRAN2 codes.

REFERENCES

[1] Balana, A. R., D. Gross, and D. R. Miller (1982). Analytical models for transient behavior of repairable item provisioning. Technical Paper Serial T-462, Institute for Management Science and Engineering, The George Washington University.

[2] Grassmann, W. (1977). Transient solutions in Markovian queueing systems, *Computers & Operations Research*, Vol. 4, pp. 47-56.

[3] Gross, D., L. Kioussis, and D. R. Miller (1982). Analytical models for transient behavior of multi-echelon repairable item provisioning systems with finite calling populations and finite repair capacities. Technical Paper Serial T-466, Institute for Management Science and Engineering, The George Washington University.

[4] Gross, D. and D. R. Miller (1982). The randomization technique as a modeling tool and solution procedure for transient Markov processes. Technical Paper Serial T-467, Institute for Management Science and Engineering, The George Washington University.

[5] Gross, D., D. R. Miller, and R. M. Soland (1981). A closed queueing network model for multi-echelon repairable item provisioning. Technical Paper Serial T-446, Institute for Management Science and Engineering, The George Washington University.

[6] Hillestad, R. J. (1981). Dyna-METRIC: Dynamic multi-echelon technique for recoverable item control. WD-911-AF, Rand, Santa Monica, CA 90406.

[7] Hillestad, R. J. and M. J. Carrillo (1980). Models and techniques for recoverable item stockage when demand and the repair process are nonstationary--Part I: Performance measurement. N-1482-AF, Rand, Santa Monica, CA 90406.

[8] Ince, J. R. (1978). Spares provisioning for repairable items. D.Sc. dissertation, The George Washington University.

[9] Maron, M. J. (1982). *Numerical Analysis--A Practical Approach.* Macmillan, New York.

[10] Muckstadt, J. A. (1973). A model for a multi-item, multi-echelon, multi-indenture inventory system. *Management Science*, Vol. 20, pp. 472-481.

[11] Sherbrooke, C. C. (1968). METRIC: A multi-echelon technique for recoverable item control. *Operations Research*, Vol. 16, pp. 122-141.

THE GEORGE WASHINGTON UNIVERSITY

Program in Logistics

Distribution List for Technical Papers

The George Washington University
Office of Sponsored Research
Gelman Library
Vice President H. F. Bright
Dean Harold Liebowitz
Dean Henry Solomon

ONR
Chief of Naval Research
(Codes 200, 434)
Resident Representative

OPNAV
OP-40
DCNO, Logistics
Navy Dept Library
NAVDATA Automation Cmd

Naval Aviation Integrated Log Suppo

NARDAC Tech Library

Naval Electronics Lab Library

Naval Facilities Eng Cmd Tech Library

Naval Ordnance Station
Louisville, Ky.
Indian Head, Md.

Naval Ordnance Sys Cmd Library

Naval Research Branch Office
Boston
Chicago
New York
Pasadena
San Francisco

Naval Ship Eng Center
Philadelphia, Pa.

Naval Ship Res & Dev Center

Naval Sea Systems Command
PMS 30611
Tech Library
Code 073

Naval Supply Systems Command
Library
Operations and Inventory Analysis

Naval War College Library
Newport

BUPERS Tech Library

FMSO

USN Ammo Depot Earle

USN Postgrad School Monterey
Library
Dr Jack R. Borsting
Prof C. R. Jones

US Coast Guard Academy
Capt Jimmie D. Woods

US Marine Corps
Commandant
Deputy Chief of Staff, R&D

Marine Corps School Quantico
Landing Force Dev Ctr
Logistics Officer

Armed Forces Industrial College

Armed Forces Staff College

Army War College Library
Carlisle Barracks

Army Cmd & Gen Staff College

Army Logistics Mgt Center
Fort Lee

Commanding Officer, USALDSRA
New Cumberland Army Depot

Army Inventory Res Ofc
Philadelphia

Army Trans Material Cmd
TCMAC-ASDT

Air Force Headquarters
AFADS-3
LEXY
SAF/ALG

Griffiss Air Force Base
Reliability Analysis Center

Gunter Air Force Base
AFLMC/XR

Maxwell Air Force Base Library

Wright-Patterson Air Force Base
AFLC/OA
Research Sch Log
AFALD/XR

Defense Technical Info Center

National Academy of Sciences
Maritime Transportation Res Bd Lib

National Bureau of Standards
Dr B. H. Colvin
Dr Joan Rosenblatt

National Science Foundation

National Security Agency

Weapons Systems Evaluation Group

British Navy Staff

National Defense Hdqtrs, Ottawa
Logistics, OR Analysis Estab

American Power Jet Co
George Chernowitz

General Dynamics, Pomona

General Research Corp
Library

Logistics Management Institute
Dr Murray A. Geisler

Rand Corporation
Library
Mr William P. Hutzler

Carnegie-Mellon University
Dean H. A. Simon
Prof G. Thompson

Case Western Reserve University
Prof B. V. Dean
Prof M. Mesarovic

Cornell University
Prof R. E. Bechhofer
Prof R. W. Conway
Prof Andrew Schultz, Jr.

Cowles Foundation for Research
in Ecomonics
Prof Martin Shubik

Florida State University
Prof R. A. Bradley

Harvard University
Prof W. G. Cochran
Prof Arthur Schleifer, Jr.

Princeton University
Prof A. W. Tucker
Prof J. W. Tukey
Prof Geoffrey S. Watson

Purdue University
Prof S. S. Gupta
Prof H. Rubin
Prof Andrew Whinston

Stanford University
Prof T. W. Anderson
Prof Kenneth Arrow
Prof G. B. Dantzig
Prof F. S. Hillier
Prof D. L. Iglehart
Prof Samuel Karlin
Prof G. J. Lieberman
Prof Herbert Solomon
Prof A. F. Veinott, Jr.

University of California, Berkeley
Prof R. E. Barlow
Prof D. Gale
Prof Jack Kiefer

University of California, Los Angeles
Prof R. R. O'Neill

University of North Carolina
Prof W. L. Smith
Prof M. R. Leadbetter

University of Pennsylvania
Prof Russell Ackoff

University of Texas
Institute for Computing Science
and Computer Applications

Yale University
Prof F. J. Anscombe
Prof H. Scarf

Prof Z. W. Birnbaum
University of Washington

Prof B. H. Bissinger
The Pennsylvania State University

Prof Seth Bonder
University of Michigan

Prof G. E. Box
University of Wisconsin

Dr Jerome Bracken
Institute for Defense Analyses

Continu

Prof A. Charnes
University of Texas

Prof H. Chernoff
Mass Institute of Technology

Prof Arthur Cohen
Rutgers - The State University

Mr Wallace M. Cohen
US General Accounting Office

Prof C. Derman
Columbia University

Prof Masao Fukushima
Kyoto University

Prof Saul I. Gass
University of Maryland

Dr Donald P. Gaver
Carmel, California

Prof Amrit L. Goel
Syracuse University

Prof J. F. Hannan
Michigan State University

Prof H. O. Hartley
Texas A & M Foundation

Prof W. M. Hirsch
Courant Institute

Dr Alan J. Hoffman
IBM, Yorktown Heights

Prof John R. Isbell
SUNY, Amherst

Dr J. L. Jain
University of Delhi

Prof J. H. K. Kao
Polytech Institute of New York

Prof W. Kruskal
University of Chicago

Mr S. Kumar
University of Madras

Prof C. E. Lemke
Rensselaer Polytech Institute

Prof Loynes
University of Sheffield, England

Prof Tom Maul
Kowloon, Hong Kong

Prof Steven Nahmias
University of Santa Clara

Prof D. B. Owen
Southern Methodist University

Prof P. R. Parathasarathy
Indian Institute of Technology

Prof E. Parzen
Texas A & M University

Prof H. O. Posten
University of Connecticut

Prof R. Remage, Jr.
University of Delaware

Prof Hans Riedwyl
University of Berne

Mr David Rosenblatt
Washington, D. C.

Prof M. Rosenblatt
University of California, San Diego

Prof Alan J. Rowe
University of Southern California

Prof A. H. Rubenstein
Northwestern University

Prof Thomas L. Saaty
University of Pittsburgh

Dr M. E. Salveson
West Los Angeles

Prof Gary Scudder
University of Minnesota

Prof Edward A. Silver
University of Waterloo, Canada

Prof Rosedith Sitgreaves
Washington, DC

LTC G. L. Slyman, MSC
Department of the Army

Prof M. J. Sobel
Georgia Inst of Technology

Prof R. M. Thrall
Rice University

Dr S. Vajda
University of Sussex, England

Prof T. M. Whitin
Wesleyan University

Prof Jacob Wolfowitz
University of South Florida

Prof Max A. Woodbury
Duke University

Prof S. Zacks
SUNY, Binghamton

Dr Israel Zang
Tel-Aviv University

February 1981